

MangoPi Guide for Setting Up LVGL

Prepare the repo:

Get the latest repo with all the latest BSP: <https://github.com/RT-Thread/rt-thread.git> this is the repo I used. (My case I have saved the repo at the location: /rtsmart/rthread-smart/userapps)

Also update the toolchain to latest using the command from the tools folder.

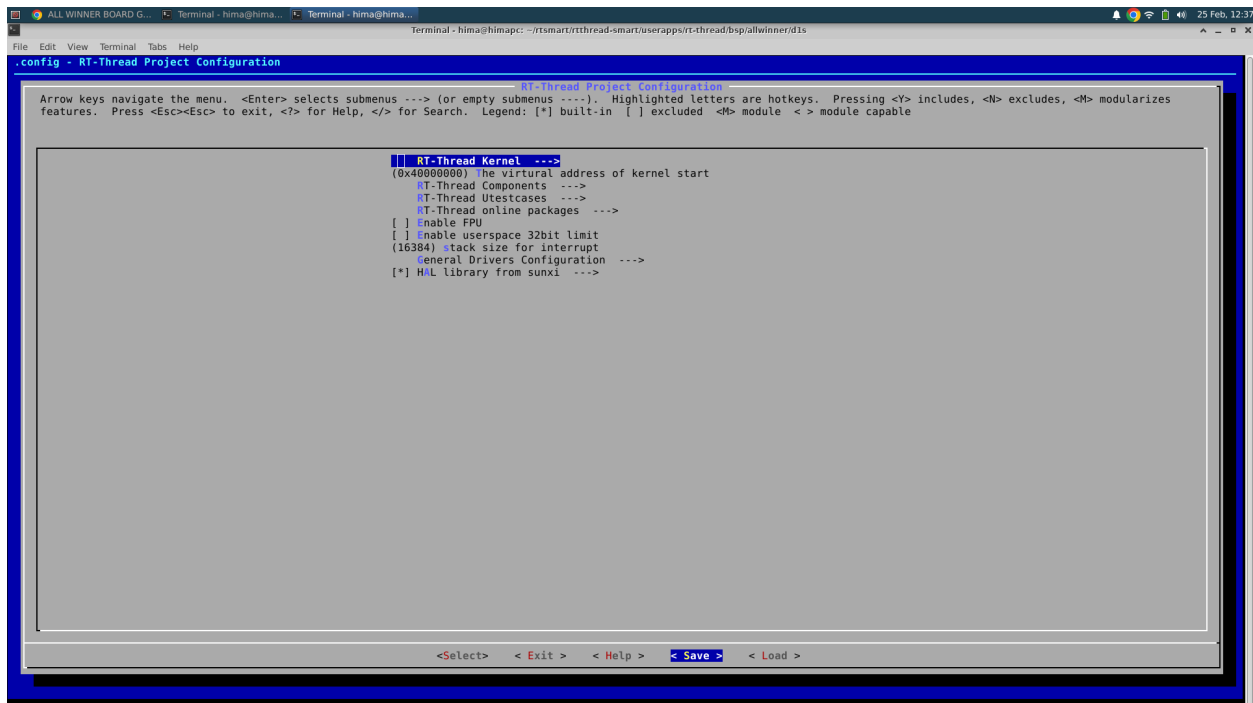
```
~/rtsmart/rthread-smart/tools$ python3 get_toolchain.py riscv64
```

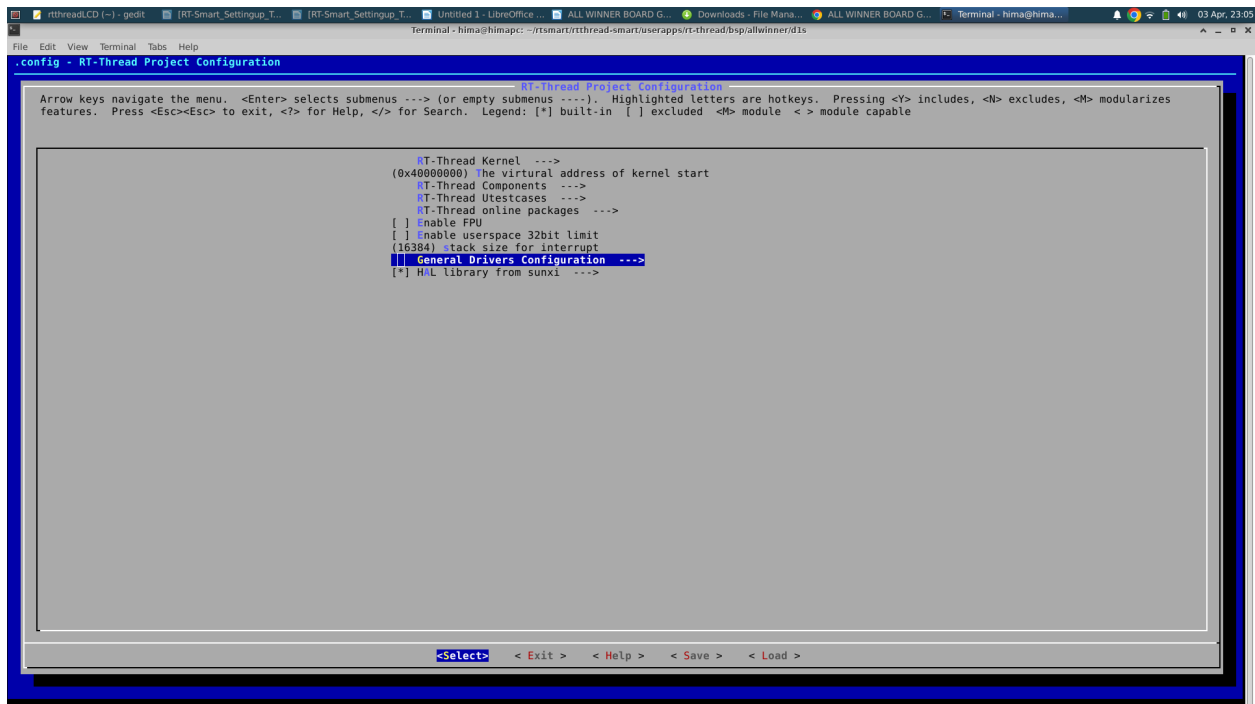
Also make sure that the environment is set correctly. `source ./smart-env.sh riscv64` must be executed at the correct folder.

Build the BSP with LCD Enabled:

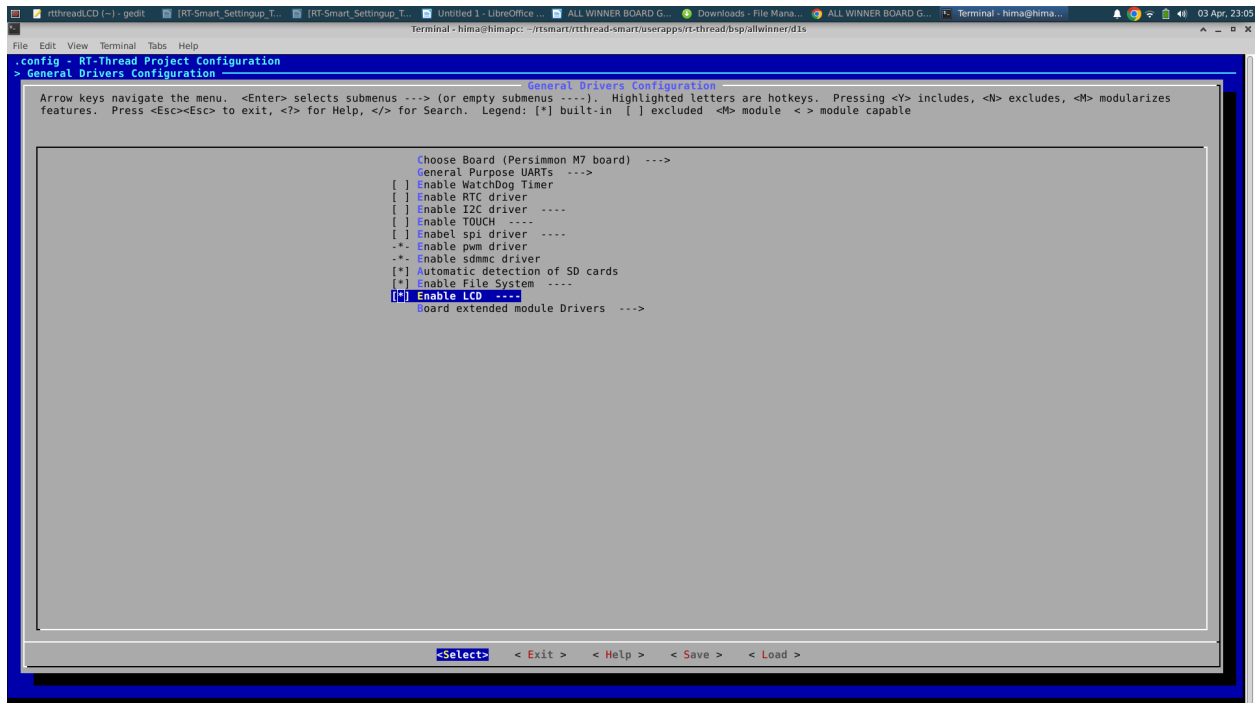
Build the BSP for the board: Now build the right BSP for your allwinner board. Navigate to the folder , in my case rtsmart/rthread-smart/userapps/rt-thread/bsp/allwinner/d1s

```
/rtsmart/rthread-smart/userapps/rt-thread/bsp/allwinner/d1s$ scones --menuconfig
```

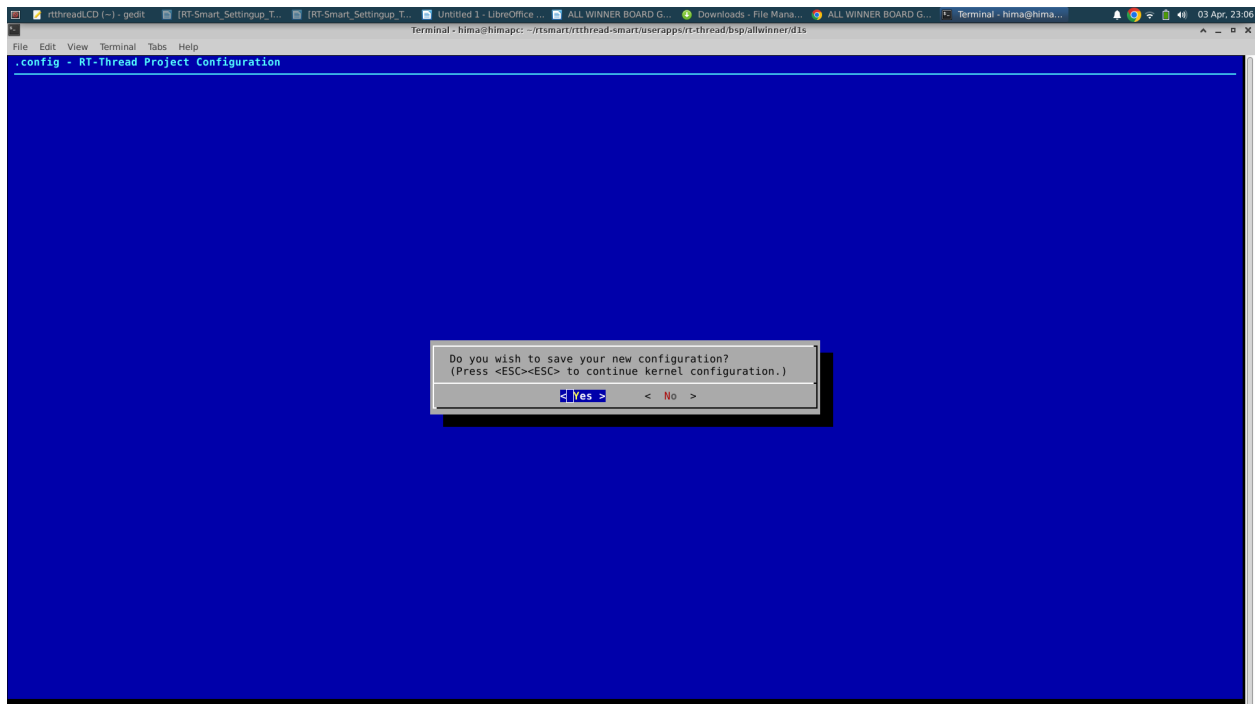




Select the “General Driver Configuration” Option



Select the “Enable LCD” optiion. To enable LCD you need press the key “Y”.



For exiting the screen you need to press <Esc> <Esc> and repeat the same step again. Then you will get a prompt for saving the configuration as shown in the image. Select "Yes" to save the file. Now LCD is enabled in the BSP. BSP should be built after this.

Execute the command "scons" to build the BSP.

```
~/rtsmart/rththread-smart/userapps/rt-thread/bsp/allwinner/d1s$ scons
.....
.....
scons: done building targets.
```

Check for and sd.bin file which gets generated. This is the kernel image we are going to store in the SD Card, which will boot the RT-Thread kernel.

Preparing SD Card:

This step needs to be done only once. If the SD Card is already formatted no need to repeat this process.

Connect the SD Card to your system. Find out the entry corresponding to your SD Card from /dev folder. For me the name was sdb. Format the sdb disk using fdisk command.

```
:~$ sudo fdisk /dev/sdb
Welcome to fdisk (util-linux 2.34).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

Command (m for help): o
Created a new DOS disklabel with disk identifier 0xbf06dddf.

Command (m for help): n
Partition type
  p   primary (0 primary, 0 extended, 4 free)
  e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-125542399, default 2048): 16384
Last sector, +/-sectors or +/-size{K,M,G,T,P} (16384-125542399, default 125542399):

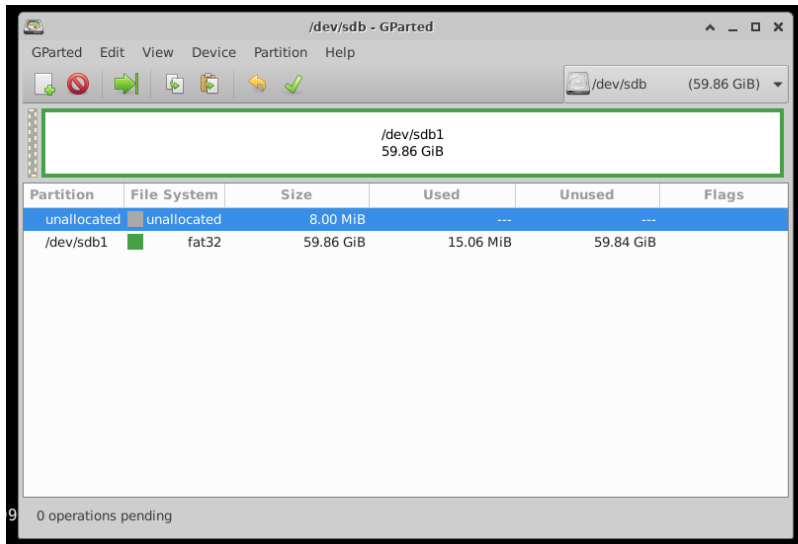
Created a new partition 1 of type 'Linux' and of size 59,9 GiB.
Partition #1 contains a vfat signature.

Do you want to remove the signature? [Y]es/[N]o: Yes

The signature will be removed by a write command.

Command (m for help): q
```

You can use *gparted* command to cross verify the partition.



Flashing Images to SD Card

Now Flash the sd.bin and boot image onto the SDCard. Use the following two commands from the right folder. Boot file is present inside the *rt-thread/bsp/allwinner/d1s/tools* folder and the sd.bin gets generated in the *rt-thread/bsp/allwinner/d1s* folder. If the boot file is already flashed then it need not be repeated. The sd.bin file needs to be flashed whenever there is a change in the kernel. In this case we have changes in kernel, so we need to flash the sd.bin file.

```
sudo dd if=boot0_sdcard_sun20iw1p1_d1s.bin of=/dev/sdb bs=1024 seek=8 (need to be done only once)
sudo dd if=sd.bin of=/dev/sdb bs=1024 seek=56
```

Command logs are followed.

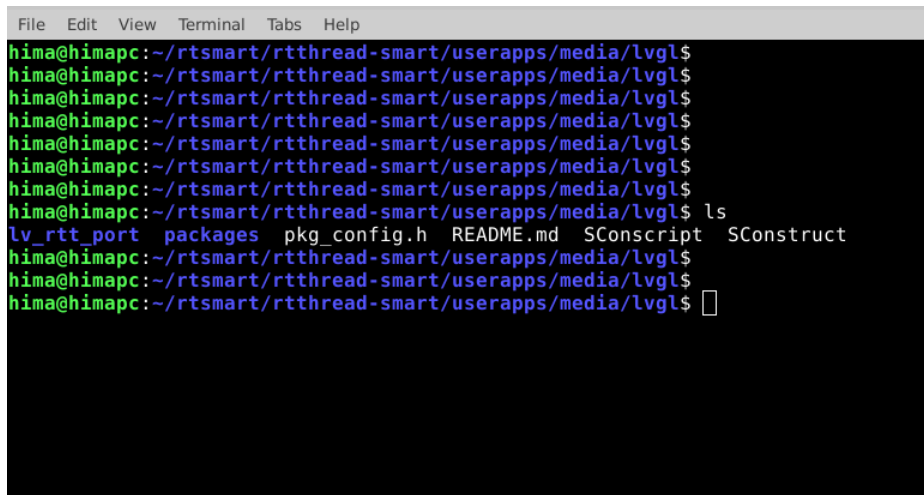
```
hima@himapc:~/rtsmart/rththread-smart/userapps/rt-thread/bsp/allwinner/d1s/tools$ sudo dd
if=boot0_sdcard_sun20iw1p1_d1s.bin of=/dev/sdb bs=1024 seek=8
48+0 records in
48+0 records out
49152 bytes (49 kB, 48 KiB) copied, 0,0181441 s, 2,7 MB/s
hima@himapc:~/rtsmart/rththread-smart/userapps/rt-thread/bsp/allwinner/d1s/tools$
hima@himapc:~/rtsmart/rththread-smart/userapps/rt-thread/bsp/allwinner/d1s/tools$
hima@himapc:~/rtsmart/rththread-smart/userapps/rt-thread/bsp/allwinner/d1s/tools$ cd ..
hima@himapc:~/rtsmart/rththread-smart/userapps/rt-thread/bsp/allwinner/d1s$ sudo dd
if=sd.bin of=/dev/sdb bs=1024 seek=56
```

```
772+0 records in
772+0 records out
790528 bytes (791 kB, 772 KiB) copied, 0,152477 s, 5,2 MB/s
hima@himapc:~/rtsmart/rththread-smart/userapps/rt-thread/bsp/allwinner/d1s$
```

Build The User Application

The LVGL stands for light and Versatile Embedded Graphics Library. A sample application which shows graphics on the LCD, just to use as the demo is already available in this git repo: <https://github.com/Rbb666/RT-Smart-UserAPP>

We are using this application. So download the repo to the folder userapps/media/lvgl



```
File Edit View Terminal Tabs Help
hima@himapc:~/rtsmart/rththread-smart/userapps/media/lvgl$
hima@himapc:~/rtsmart/rththread-smart/userapps/media/lvgl$
hima@himapc:~/rtsmart/rththread-smart/userapps/media/lvgl$
hima@himapc:~/rtsmart/rththread-smart/userapps/media/lvgl$
hima@himapc:~/rtsmart/rththread-smart/userapps/media/lvgl$
hima@himapc:~/rtsmart/rththread-smart/userapps/media/lvgl$
hima@himapc:~/rtsmart/rththread-smart/userapps/media/lvgl$
hima@himapc:~/rtsmart/rththread-smart/userapps/media/lvgl$ ls
lv_rtt_port  packages  pkg_config.h  README.md  SConscript  SConstruct
hima@himapc:~/rtsmart/rththread-smart/userapps/media/lvgl$
hima@himapc:~/rtsmart/rththread-smart/userapps/media/lvgl$
hima@himapc:~/rtsmart/rththread-smart/userapps/media/lvgl$
```

Now we need to edit Sconstruct file so that the lvgl application is also build. For that edit to the userapps/Sconstruct file and add the lvgl folder also to it , as shown in the following image

```

16     for item in list:
17         path = os.path.join(dir, item)
18         if os.path.isfile(os.path.join(path, 'SConscript')):
19             BuildApplication(item, path + '/SConscript', usr_root = '.')
20
21 AddOption('--app',
22           dest = 'make-application',
23           type = 'string',
24           default = None,
25           help = 'make application')
26
27 AddOption('--dir',
28           dest = 'make-in-directory',
29           type = 'string',
30           default = None,
31           help = 'make in directory')
32
33 AddOption('--sdk-libc',
34           dest='sdk-libc',
35           action='store_true',
36           default=False,
37           help='build with sdk libc')
38
39 if GetOption('sdk-libc'):
40     AddDepend('SDK_LIBC')
41
42 if GetOption('make-application'):
43     item = GetOption('make-application')
44
45     path = os.path.join(cwd, 'apps', item)
46     if os.path.isfile(os.path.join(path, 'SConscript')):
47         BuildApplication(item, path + '/SConscript', usr_root = '.')
48 elif GetOption('make-in-directory'):
49     dir = GetOption('make-in-directory')
50     BuildDir(os.path.join(cwd, dir))
51 else:
52     BuildDir(os.path.join(cwd, 'apps'))
53     # BuildDir(os.path.join(cwd, 'testcases'))
54     BuildDir(os.path.join(cwd, 'services'))
55     BuildDir(os.path.join(cwd, 'media'))

```

We

are ready with the user apps. Now clean and build the user applications. Remember to run the smart-env.sh prior to the building.

```

~/rtsmart/rththread-smart$ source smart-env.sh riscv64
Arch    => riscv64
CC      => gcc
PREFIX  => riscv64-unknown-linux-musl-
EXEC_PATH => /home/hima/rtsmart/rththread-smart/tools/gnu_gcc/riscv64-linux-
musleabi_for_x86_64-pc-linux-gnu/bin

~/rtsmart/rththread-smart/userapps$ sconsclean --clean
sconsclean: Reading SConscript files ...
sconsclean: done reading SConscript files.
sconsclean: Cleaning targets ...
Removed build/hello/main.o
Removed build/lvgl/lvgl_rtt_port/lvgl_demo.o
Removed build/lvgl/lvgl_rtt_port/lvgl_port_disp.o

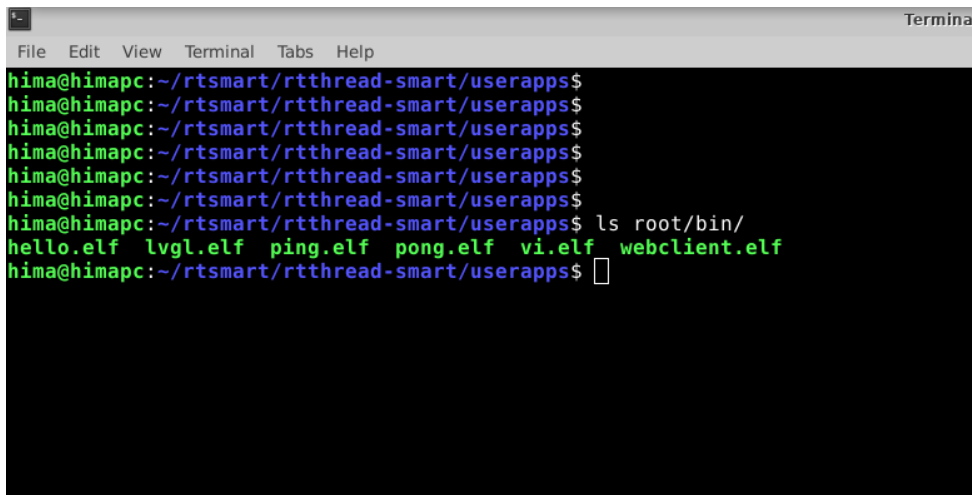
.....

~/rtsmart/rththread-smart/userapps$ sconsclean
sconsclean: Reading SConscript files ...
sconsclean: done reading SConscript files.
sconsclean: Building targets ...
CC build/hello/main.o
CC build/lvgl/lvgl_rtt_port/lvgl_demo.o

```

```
CC build/lvgl/lv_rtt_port/lv_port_disp.o
.....
CC media/lvgl/packages/LVGL-latest/src/widgets/win/lv_win.o
LINK root/bin/hello.elf
LINK root/bin/lvgl.elf
LINK root/bin/ping.elf
LINK root/bin/pong.elf
LINK root/bin/vi.elf
scons: done building targets.
```

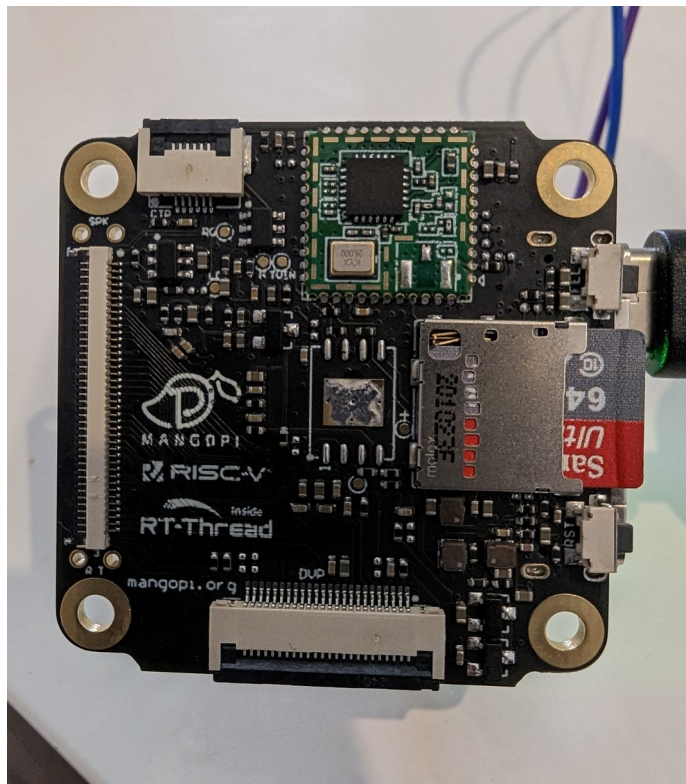
The generated elf files are located in the root/bin folder. We need to copy this bin folder also to the SD card. This can be just manually copied.



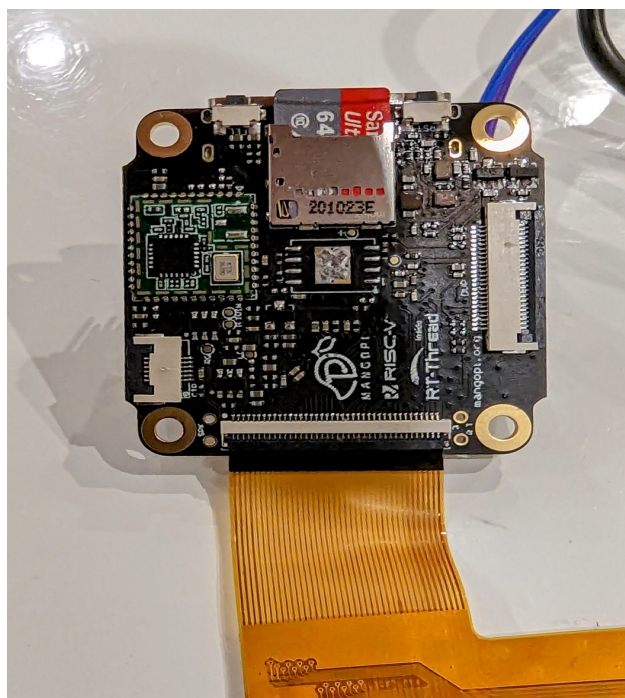
```
hima@himapc:~/rtsmart/rcthread-smart/userapps$
hima@himapc:~/rtsmart/rcthread-smart/userapps$
hima@himapc:~/rtsmart/rcthread-smart/userapps$
hima@himapc:~/rtsmart/rcthread-smart/userapps$
hima@himapc:~/rtsmart/rcthread-smart/userapps$
hima@himapc:~/rtsmart/rcthread-smart/userapps$ ls root/bin/
hello.elf  lvgl.elf  ping.elf  pong.elf  vi.elf  webclient.elf
hima@himapc:~/rtsmart/rcthread-smart/userapps$
```

Setting Up The Hardware

Plug the SD Card on to the allwinner Board, on the microSD Card slot.



Connect the LCD Module to the MangoPi board. I am using GL043026 Graphics LCD module. The connection is shown in the following figure.

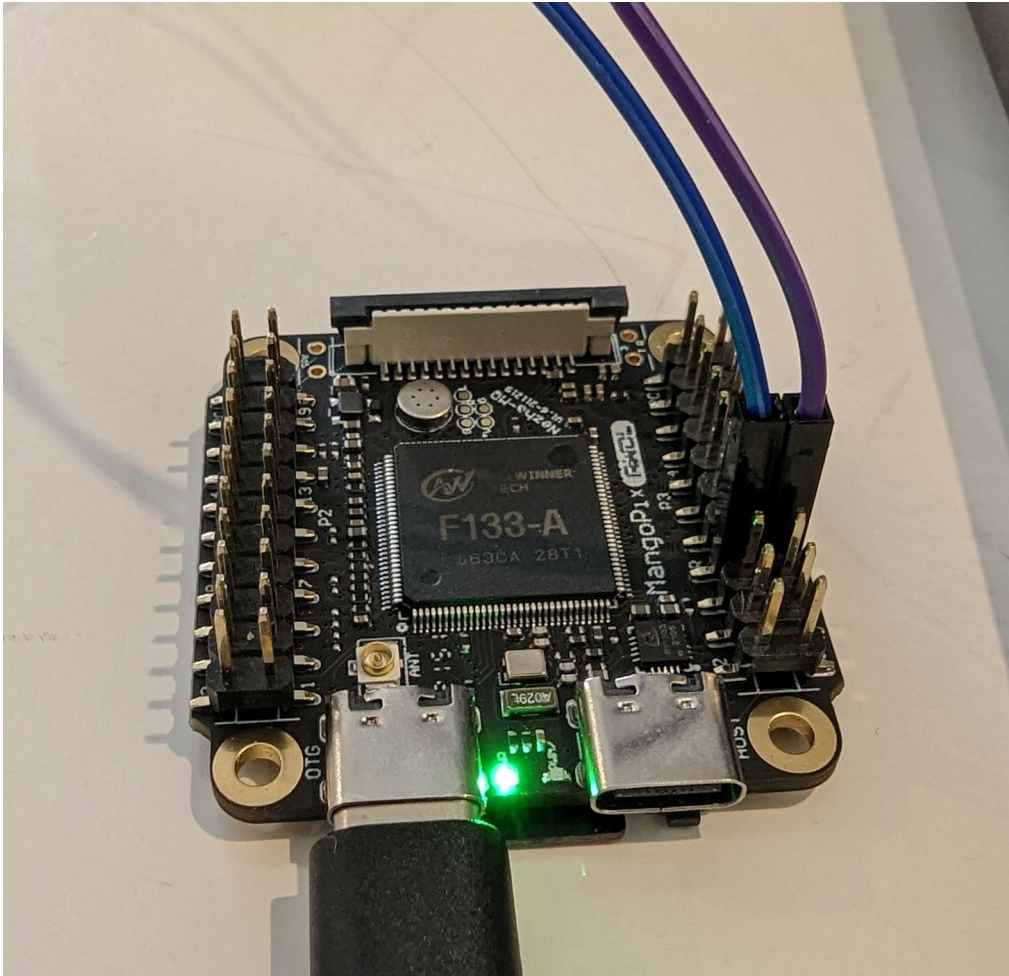


Also connect a UART-USB Bridge, so that you can check the UART Commands from your PC. I am using the CP2102 USB to TTL interface board as UART-USB Bridge.

RX (CP2102) → TX (P3.7) Allwinner

TX (CP2102) → RX(P3.8) Allwinner

Refer to the following figure.



Check the UART messages on a suitable Terminal Application.

Baud Rate: 500000

Open any Serial port terminal check if the board is booting properly. If everything works fine you can see the following messages on the serial port.

```
[00:02:21:190] [854]run_addr = 0x40200000%r
[00:02:21:190] [855]index = 0x0%r
[00:02:21:190] [856]Entry_end = 0x3b454949%r
[00:02:21:199] [857]*****TOC1 Item Message*****%r
[00:02:21:199] [860]Entry_name = kernel%r
[00:02:21:199] [860]Entry_data_offset = 0x20600%r
[00:02:21:199] [861]Entry_data_len = 0xe69c0%r
[00:02:21:199] [862]encrypt = 0x0%r
[00:02:21:199] [863]Entry_type = 0x0%r
[00:02:21:199] [863]run_addr = 0x40400000%r
[00:02:21:199] [864]index = 0x0%r
[00:02:21:206] [865]Entry_end = 0x3b454949%r
[00:02:21:206] [865]*****%r
[00:02:21:206] [868]Jump to second Boot.%r
[00:02:21:223] %r
[00:02:21:224] OpenSBI v0.9-167-gf45b7d4%r
[00:02:21:224] %r
[00:02:21:230] %r
[00:02:21:230] %r
[00:02:21:230] %r
[00:02:21:230] %r
[00:02:21:230] %r
[00:02:21:230] %r
[00:02:21:236] %r
[00:02:21:236] %r
[00:02:21:245] Platform Name : Allwinner D1 NeZha%r
[00:02:21:246] Platform Features : medeleg%r
[00:02:21:251] Platform HART Count : 1%r
[00:02:21:251] Platform IPI Device : aclint-mswi%r
[00:02:21:251] Platform Timer Device : aclint-mtimer @ 24000000Hz%r
[00:02:21:251] Platform Console Device : uart8250%r
[00:02:21:251] Platform HSM Device : ---%r
[00:02:21:251] Platform Reboot Device : sunxi-wdt-reset%r
[00:02:21:258] Platform Shutdown Device : ---%r
[00:02:21:258] %r
Clear Hex output Logging to: /root/cutecom.log
Device: /dev/ttyUSB0 Connection: 500000 @ 8-N-1
```

Now execute the bin/lvgl.elf application.

```
[00:02:22:048] sdmmc bytes_per_secotr:200, sector count:77ba000%  
[00:02:22:059] found part[0], begin: 8388608, size: 59.876GB%  
[00:02:22:059] found partition:sd0 of mbr at offset 0x0000000000004000, size:0x00000000077b6000%  
[00:02:22:059] hal_sd_create 1%  
[00:02:22:059] card_detect insert%  
[00:02:22:092] Initial card failed!%  
[00:02:22:093] [E/drv-sdmmc] init sdmmc failed!%  
[00:02:22:099] [E/drv-sdmmc] sdmmc_init failed!%  
[00:02:22:099] [D/FAL] (fal_flash_init:47) Flash device |          sdcard0 | addr: 0x00000000 | len: 0xf7400000 | blk_size: 0x00000200 | initialized finish.%  
[00:02:22:099] <0x1b>[32:22m|/FAL] ===== FAL partition table =====<0x1b>[0m%  
[00:02:22:105] <0x1b>[32:22m|/FAL] | name | flash_dev | offset | length |<0x1b>[0m%  
[00:02:22:105] <0x1b>[32:22m|/FAL] -----<0x1b>[0m%  
[00:02:22:105] <0x1b>[32:22m|/FAL] | download | sdcard0 | 0x00800000 | 0x00800000 |<0x1b>[0m%  
[00:02:22:105] <0x1b>[32:22m|/FAL] | easyflash | sdcard0 | 0x01000000 | 0x00100000 |<0x1b>[0m%  
[00:02:22:109] <0x1b>[32:22m|/FAL] | filesystem | sdcard0 | 0x01100000 | 0x00c00000 |<0x1b>[0m%  
[00:02:22:109] <0x1b>[32:22m|/FAL] =====<0x1b>[0m%  
[00:02:22:114] <0x1b>[32:22m|/FAL] RT-Thread Flash Abstraction Layer initialize success.<0x1b>[0m%  
[00:02:22:114] Hello RISC-V%  
[00:02:22:114] [W/DBG] disp:[parser_disp_init_para 575]of_property_read screen1_output_type fail%  
[00:02:22:120] %  
[00:02:22:120] %  
[00:02:22:120] msh />Mount "sd0p0" on "/" success%  
[00:02:24:804] %  
[00:02:24:811] msh />ls bin/%  
[00:02:31:085] Directory bin:%  
[00:02:31:092] webclient.elf 391856 %  
[00:02:31:092] hello.elf 340064 %  
[00:02:31:092] lvgl.elf 3935488 %  
[00:02:31:092] ping.elf 346168 %  
[00:02:31:092] pong.elf 340992 %  
[00:02:31:101] vi.elf 516288 %  
[00:02:31:101] msh />%  
[00:02:33:763] msh />bin/lvgl.elf%  
[00:02:42:331] msh />
```

Clear Hex output Logging to: /root/cutecom.log
Device: /dev/ttyUSB0 Connection: 500000 @ 8-N-1

Upon execution, you can see a video being played on the graphics LCD Screen.

